VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2016 Petr Řepecký

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student:	**Petr Řepecký**
Studijní program:	B2647 Informační a komunikační technologie
Studijní obor:	2612R025 Informatika a výpočetní technika
Téma:	Absolvování individuální odborné praxe Individual Professional Practice in the Company
Jazyk vypracování:	čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: proHR leaders s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Moravec, Ph.D.**

Konzultant bakalářské práce: Ing. Lubomír Urbánek

Datum zadání: 01.09.2015
Datum odevzdání: 29.04.2016

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 29. dubna 2016 .....

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

proHR
proHR leaders s.r.o.
Nádražní 344/23,150 00 Praha 5
DIČ: CZ26968584 www.phrl.cz

V Ostravě 29. dubna 2016

Tímto bych rád poděkoval především Ing. Lubomíru Urbánkovi za možnost vykonat odbornou praxi ve firmě proHR Leaders s.r.o a dále pak za jeho skvělý přístup, ochotu a vstřícnost. Dále pak patří mé díky celému pracovnímu týmu, který se podílel na vývoji aplikace proHR za odborné rady a pomoc, kterou mi během odborné praxe poskytl. Nakonec bych také rád poděkoval svému vedoucímu bakalářské práce Ing. Pavlu Moravcovi, Ph.D., za vstřícnost a rady při psaní této práce.

Abstrakt

Tématem této bakalářské práce bude popis a zhodnocení mé práce v rámci odborné praxe, kterou jsem vykonával ve firmě proHR Leaders s.r.o, zaměřenou na lidské zdroje a pracovní procesy. V rámci zaměření proHR Leaders za spolupráce se specializovanou Ostravskou firmou vyvíjí webovou aplikaci, která má zefektivnit firmám klientů přehled o zaměstnancích, jejich úkolech a umožnit uplatnit zaměstnaneckou zpětnou vazbu. Práce na vývoji této aplikace byla náplní mé praxe a jsou v ní popsány úkoly, na kterých jsem pracoval, jak jsem je řešil a jaké nové znalosti o technologiích a pracovních postupech jsem získal.

Klíčová slova: proHR Leaders, proHR, CSS, HTML, JavaScript, PHP, Bootstrap, KingAdmin, Zend Framework, Highcharts

Abstract

The topic of this bachelor thesis is a description and summary of my work within practical training, which was performed in the proHR Leaders company, focusing on human resources and work processes. In cooperation with specialized company in Ostrava, proHR Leaders is developing a web application that has to improve overview of the employees, their tasks and enable to provide employee feedback. Work on the development of this application was the focus of my practice, and I will describe the tasks, which I worked on, how I dealt with them and what new knowledge about technologies and workflows I've got.

Key Words: proHR Leaders, proHR, CSS, HTML, JavaScript, PHP, Bootstrap, KingAdmin, Zend Framework, Highcharts

Obsah

Seznam použitých zkratek a symbolů

MVC	–	Model-View-Controller
PHP	–	Hypertext Preprocessor
HTML	–	HyperText Markup Language
OOP	–	Objektově orientované programování
CSS	–	Cascading Style Sheets
MSSQL	–	Microsoft Structured Query Language
SQL	–	Structured Query Language

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Jako bakalářskou práci jsem si vybral odbornou praxi ve firmě na pozici programátora webových aplikací. Odbornou praxi jsem si zvolil místo bakalářské práce na téma, jelikož praxi považuji za nejlepší zdroj zkušeností, nových znalostí a měl jsem díky ní možnost si poprvé vyzkoušet práci v oboru. Praxi jsem vykonal ve firmě proHR Leaders s.r.o, ve které jsem se podílel na implementaci a testování webové aplikace, která má po dokončení sloužit k evidenci výkonu zaměstnanců v pracovním procesu a návrhu možností na jejich zlepšení.

1.1 Popis firmy

Firma proHR Leader s.r.o je zaměřená na výzkum lidských zdrojů a pracovních procesů. V současné době je ve spolupráci se specializovanou Ostravskou firmou vyvíjena webová aplikace nazvaná proHR, která klientům umožní evidovat výkony zaměstnanců v pracovních procesech a dále pomáhat je zlepšovat. Zatímco firma proHR Leaders s.r.o navrhuje vylepšení, nové funkce a moduly, tak specializovaná firma obstarává programování a samotný vývoj. Firma, pod jejíž záštitou se aplikace vyvíjí, se zaměřuje na vývoj a programování webových stránek a aplikací, dále pak na procesní analýzu a optimalizaci výkonu aplikací.

Obrázek 1: Logo společnosti

1.2 Pracovní zařazení

Ve firmě jsem byl zařazen na pozici programátora. Byl jsem obeznámen s pracovní náplní, s technologiemi, které jsou využívány, a které mám nastudovat a dále pak jsem dostal přístup na server firmy abych měl přístup ke zdrojovým kódům aplikace, na které jsem později pracoval. Se svým nadřízeným jsem se pak dále dohodnul na řešení organizačních věcí prostřednictvím schůzek ve firmě, které probíhaly, pokud přibyly požadavky na práci nebo jsem potřeboval odbornou radu.

1.3 Uplatněné znalosti získané během studia

Již na počátku mé práce na novém submodulu pro hodnocení jsem využil četné znalosti databázových systémů, které jsem získal během studia. Neměl jsem žádný větší problém při návrhu a vytváření databázových tabulek, jejich vztahů a atributů. Dále bych rád uvedl, že základní získané znalosti o architektuře *MVC* mi dopomohly dříve se zorientovat v aplikaci proHR, a dále díky pokročilým znalostem a praktickým dovednostem s *OOP* bylo studium frameworku Zend snazší. Jako poslední bych rád zmínil velmi užitečné znalosti z oblasti vývoje webových aplikací, které jsem během studia získal a které stejně jako například znalosti *JavaScriptu* mohl dále rozvíjet.

1.4 Chybějící znalosti

Jediná kompletně chybějící znalost, kterou bych rád zmínil, je znalost jazyka *PHP*. V době začátku odborné praxe jsem byl v oblasti jazyka *PHP* začátečníkem a byla potřeba se ho brzy naučit.

1.5 Použité technologie

Aplikace proHR je od počátku implementována především v jazyce *PHP*. Nejedná se o čistou formu, navíc je použit Zend Framework, se kterým má firma bohaté zkušenosti a využívá jej v mnoha dalších projektech. Ve firmě je aktuálně využívána verze 1.12 tohoto frameworku. Vzhledem k velkým změnám u vyšších verzí firma zvažuje užití zcela jiného frameworku, pokud se bude chtít rozhodnout pro změnu.

- PHP
  V jazyku *PHP* verze 5.6 využívající objektově orientovaný *Zend Framework* verze 1.12 je napsána většina zdrojových kódů aplikace.[1]

- Zend Framework
  Objektově orientovaný *Zend Framework* verze 1.12 využívající softwarovou architekturu MVC.[2]

- HTML
  Značkovací jazyk *HTML* verze 5 je využit k zobrazení obsahu stránek aplikace.[3]

- CSS
  Jazyk *CSS* verze 3 k úpravě zobrazení HTML dokumentů.[4]

- KingAdmin - Responsive Admin Dashboard
  Grafická šablona použitá k vytvoření responzivního vzhledu a efektivní možnosti práce s aplikací na libovolné platformě a zařízení. Šablona *KingAdmin - Responsive Admin Dashboard* je postavena na *Bootstrapu* 3.2.0.[5]

- JavaScript
  JavaScript je v aplikaci využit především pro dynamický vzhled aplikace, který dále také upravuje jazyk *CSS*. Použit také pro dynamické formuláře.[6]

- Highcharts
  Highcharts jsou knihovny napsány v čistém JavaScriptu určené k snadnému vytváření interaktivních grafů pro webové stránky nebo aplikace. Nabízí celou řadu grafů ať už jde o sloupcové, přímkové, kruhové nebo také koláčové grafy a jiné grafy.[7]

- MySQL
  Jako databázový systém je momentálně použito *MYSQL* s datovým uložištěm *MyISAM*. Do budoucna firma uvažuje o investici do *Oracle* nebo *MSSQL* serveru.[8]

2 Aplikace proHR

Webová aplikace *proHR* byla a je vyvíjena k prodeji pro firmy různých oborů i velikostí. Slouží k přehledné evidenci zaměstnanců, zaměstnanecké struktury a dále pak především k zefektivnění pracovních procesů díky systému hodnocení vedených ke zpětné vazbě, na kterou je pak dále možno navázat prostřednictvím návrhu rozvojových aktivit a školení.

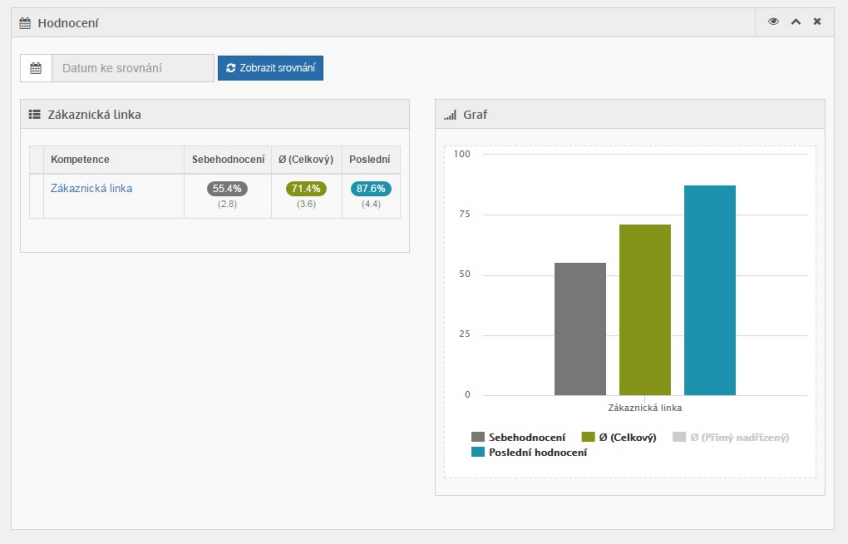
2.1 Základní struktura aplikace proHR

Webová aplikace proHR je implementována podle softwarové architektury MVC, každý modul aplikace tedy obsahuje tři základní komponenty: **model, view a controller**.

Funkcionality aplikace jsou rozdělené do modulů a submodulů, které dále vyjmenuji a popíši hlavní funkci těch nejdůležitějších.

2.1.1 Modul people

Modul **people** obsahuje veškeré funkcionality spojené s osobami v systému a je dále členěn na submoduly **user, plan, goal, team** a také **evaluation**, na kterém jsem pracoval. Zmíněné submoduly souhrnně zajišťují informace a funkce spojené s osobami, jejich cíli, hodnocením a dále pak hodnocení celých týmů zaměstnanců a plánování hodnocení zaměstnance.



Obrázek 2: Část submodulu **evaluation**

2.1.2 Modul competence

Modul **competence** se skládá z kompetenčních modelů a jejich kompetencí. Funkcí tohoto modulu je umožnit každého zaměstnance zařadit do kompetenčního modelu, který dále obsahuje různé hodnotitelné kompetence. Každý zaměstnanec tedy má přiřazenou svou pracovní pozici, v rámci které vykonává určité činnosti. Tyto činnosti pak prostřednictvím submodulu **evaluation**, patřícího do modulu **people**, lze hodnotit.

2.1.3 Modul plans

Modul **plans** umožňuje nadřízeným zaměstnancům připravovat pro své podřízené rozvojové plány, které obsahují rozvojové aktivity. Tyto aktivity jsou vázány na kompetence, ovšem rozvojové plány jsou vždy vázány pouze na jeden konkrétní kompetenční model. Cílem tohoto modulu je umožnit nadřízeným zaměstnancům reagovat na hodnocení kompetencí svých podřízených ve formě plánování rozvojových aktivit, které jejich hodnocení může dopomoci vylepšit.

2.1.4 Ostatní moduly

- **help**
  Modul pro zajišťující nápovědu pro zlepšení uživatelského komfortu.

- **login**
  Modul řešící přihlašování uživatelů do aplikace.

- **report**
  Modul pro generování **reportů**.

3 Práce na aplikaci proHR

V průběhu odborné praxe jsem pracoval na třech hlavních úkolech. Prvním z úkolů byla práce na lokalizaci, tedy zajistit vícejazyčnost webové aplikace proHR. Kromě zmíněného cíle tohoto úkolu, jsem dále měl při této příležitosti možnost se seznámit s aplikací, na které jsem po zbytek praxe pracoval, a dále pak nastudovat později potřebné znalosti. Druhým zadaným úkolem bylo vytvoření modulu pro hodnocení zaměstnanců v rámci jejich pracovního zařazení a kompetencí, které daná pozice obnáší. Třetím a posledním úkolem bylo vytvoření interaktivních grafů pro hodnocení, s využitím libovolně zvoleného frameworku.

Den	Činnost
1.	Seznámení s firmou a pracovním prostředím
1.	Zadání úkolu lokalizace, informace o znalostech k nastudování
2. - 5.	Práce na lokalizaci, studium *PHP*/*Zend*
6. - 9.	Návrh a vytvoření databázových tabulek
10. - 14.	Model **evaluation** a **evaluation_description**
15. - 20.	Model **evaluation_detail**
21. - 33.	Práce na controlleru **evaluation**
34. - 41.	Práce na view
42. - 43.	Prvotní úpravy při práci na grafech
44. - 50.	Práce na grafech

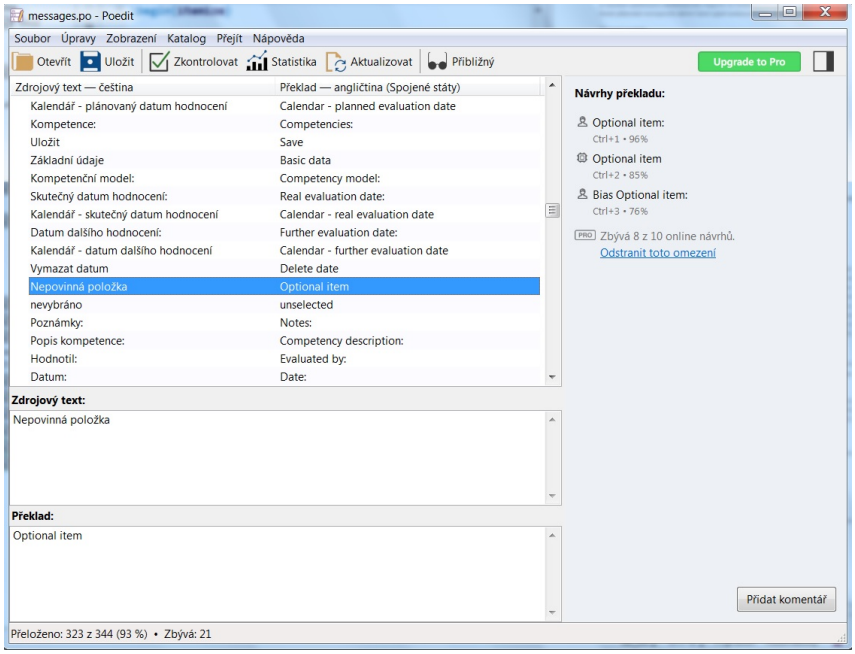
Tabulka 1: Přehled časové náročnosti jednotlivých úkolů

3.1 Lokalizace

První úkol, který mi byl během odborné praxe přidělen, byla pomocná práce na lokalizaci aplikace. V době, kdy jsem nastoupil na praxi, ještě nebyla samotná implementace lokalizace hotová a tak jsem se mohl věnovat studiu jazyka *PHP* a frameworku *Zend*. Jako začátečník v programování webových aplikací jsem měl za úkol dostudovat potřebné znalosti k možnosti pokračovat dále v práci na aplikaci *proHR*. Ve chvíli, kdy byla implementace samotného mechanismu lokalizace hotová, jsem mohl začít pracovat na přepisu zdrojových kódů, aby byla všechna slova připravena k překladu. Práci na lokalizaci považuji za skvělý začátek praxe, jelikož jsem měl možnost se plně seznámit se všemi zdrojovými kódy aplikace a získat všeobecný rozhled o jejím fungování.

3.1.1 Překlad

Překlad aplikace byl realizován pomocí dvojice souborů **.po a .mo** pro každý z překládaných jazyků. Soubory **.po** obsahují překlady a soubory **.mo** jsou zkompilovanou verzí souborů **.po**. Překlady jsou uloženy v souborech **.po** v následujícím formátu:
#: [Cesta k souboru]:[Číslo řádku] [msgid] [Identifikátor] [msgstr] [Přeložený řetězec]
Zkompilované soubory **.mo** lze dále jednoduše modifikovat a číst, například pomocí nástroje **Poedit** s kterým jsem pracoval. Program **Poedit** sám prohledává zdrojové kódy aplikace a hledá v nich výskyty nastavených klíčových slov. V tomto případu se jednalo o *PHP* funkci **gettext()**.



Obrázek 3: Rozhraní nástroje Poedit

```
//Funkce trans pracujici jako gettext

<?php echo $this->trans("Kompetencni model:"); ?>
```

Výpis 1: Ukázka překladu

3.2 Vytvoření modulu pro hodnocení zaměstnanců

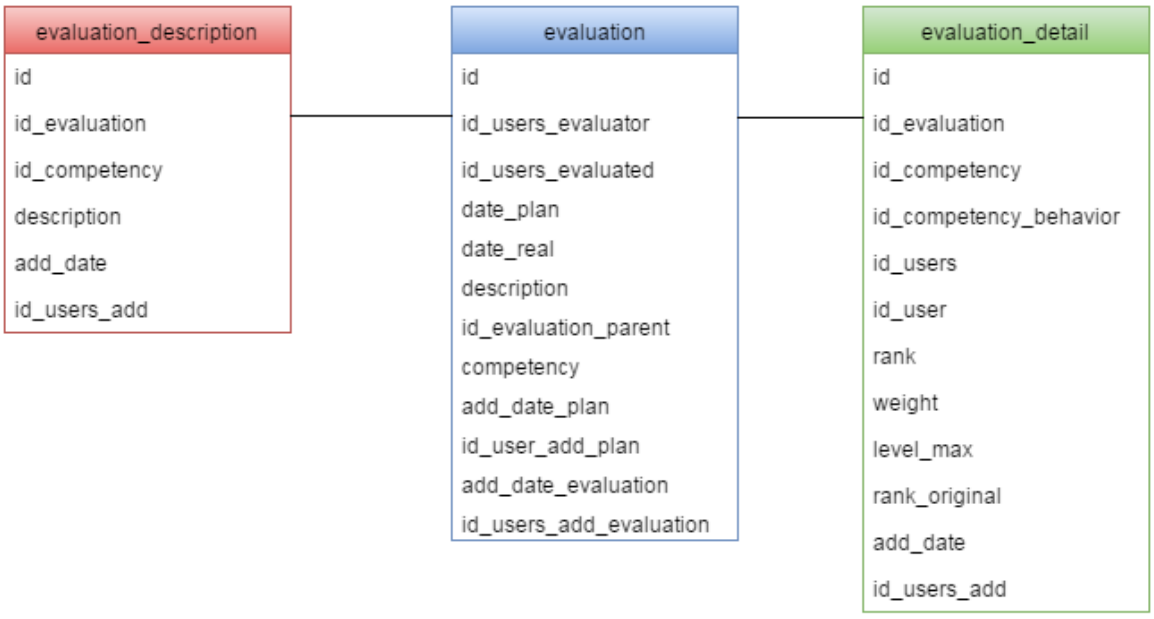
Většinu mé práce při odborné praxi tvořila implementace nového modulu pro hodnocení zaměstnanců, který dále rozšiřoval stávající modul **people**.

Modul pro hodnocení má za úkol umožnit hodnocení jednotlivých kompetencí zaměstnanců, podle jejich zařazení v kompetenčním modelu (tzn. pracovní pozice). Hodnocení je rozděleno do tří skupin.

- Hodnocení nadřízeného.
- Hodnocení pověřeného - Jedná se o hodnocení zaměstnancem, který je nadřízeným pověřen k hodnocení.
- Sebehodnocení.

3.2.1 Databáze

První fází vytváření nového modulu byl návrh a tvorba databázových tabulek a jejich atributů. Tabulky jsou v součtu tři, a spojení mezi nimi je realizováno pomocí cizích klíčů.



Obrázek 4: Vytvořené tabulky pro modul hodnocení

- **evaluation**
  Tabulka obsahující základní informace o hodnocení, jako například komu hodnocení patří, na kdy bylo hodnocení plánováno a kdy se opravdu uskutečnilo.

- **evaluation_description**
  Tabulka popisující hodnocení s atributy jako slovní popis, datum přidání a s propojením na tabulky **evaluation a competency** skrze cizí klíče. Propojení s tabulkou **competency** zajišťuje přiřazení popisu k dané kompetenci, zatímco s tabulkou **evaluation** k danému hodnocení.

- **evaluation_detail**
  Tabulka reprezentující detail hodnocení s určením, jakého výsledku zaměstnanec v dané kompetenci dosáhnul. Tabulka obsahuje atributy jako například číselný **rank** určující známku z dané kompetence, číselný atribut **level_max** určující maximální možnou známku a datum přidání. Tabulka je propojená s tabulkami evaluation a competency skrze cizí klíče, které určují ke kterému hodnocení detail patří a jakou kompetenci hodnotí.

3.2.2 Model

Poté, co jsem v databázi vytvořil potřebné tabulky, jsem začal implementovat **model** pro každou z nich. Každý **model** má za úkol pracovat s tabulkami a jejich záznamy. Základními funkcemi, které mají všechny modely společné, jsou vkládání dat do databáze a selektování dat dle vybraného **Id**.

1. **evaluation**
   Prvním modelem na kterém jsem začal pracovat byl model **evaluation**. Model **evaluation** obsahuje funkce pro selektování hodnocení, které například nebyly ještě hodnoceny, obsahují zadaného hodnotitele nebo uživatele a nebo jsou vytvořeny zaměstnancem který, je zároveň i hodnotitelem, tedy jedna-li se o sebehodnocení. Navíc **evaluation** obsahuje funkci pro aktualizaci hodnocení s ochranou proti **SQL Injekci**.

```
public function updateEvaluation($id, array $data)
{
   $where = array(
   'id = ?' => $id,
);

return $this->update($data, $where);
}
```

Výpis 2: Ochrana před SQL injection

2. **evaluation_description**
   Model **evaluation_description** obsahuje navíc pouze jednu funkci která slouží k selekci poznámky hodnocení dle zadaného **Id kompetence a hodnoceni**.

3. **evaluation_detail**
   Jedná se o nejrozsáhlejší model řešící větší množství požadavků, které jsou stejně jako u předchozích modelů řešeny pomocí funkcí, které dále popíši.

   - **Funkce** pro selekci průměrné hodnoty hodnocení pro danou kompetenci podle **Id uživatele a kompetence**.
   - **Funkce** selektující poslední hodnocení pro daného uživatele a danou kompetenci.
   - **Funkce** pro selekci všech kompetenčních modelů ve kterých byl zadaný zaměstnanec hodnocen.
   - **Funkce** pro spočítání celkového průměru hodnocení v procentech pro daného zaměstnance ve všech kompetencích.

   Kromě zmíněných funkcí model **evaluation_detail** obsahuje několik dalších funkcí, které mají podobný význam, ovšem mají jiná kritéria a jiné vstupní parametry.

```
public function getEvaluation4User($user, $status=1)
{
$user = (int)$user;
$status = (int)$status;

$select = $this->select()
->where('id_users_evaluator = ? OR id_users_evaluated = ?', $user)
->where('status = ?', $status)
->order(array('date_plan DESC', 'id DESC'));

$all = $this->fetchAll($select);

if (!$all) {
  throw new Exception("Could not find row $user");
}

  return $all->toArray();
}
```

Výpis 3: Funkce která vrací všechny řádky s hodnocením kde figuruje daný uživatel.

3.2.3 Controller

V **controlleru** bylo potřeba vytvořit několik funkcí pro vytváření a detailní zobrazení hodnocení, vytváření a editaci plánů hodnocení a také pro zobrazení přehledu hodnocení zaměstnance v rámci jeho zařazení do kompetenčního modelu. Funkce mají následující společné rysy:

- Kontrola práv přístupu uživatele.
- Kontrola vstupů a zadání povinných údajů.
- Přesměrování uživatele po dokončení události na předchozí či jinou stránku.
- Zasílaní údajů do **view**.

```
//Parametry z URL
$idEvaluation = $this->_getParam('evaluation', 0);

//Dle parametru urcime o jake naplanovane hodnoceni se jedna
$evaluationList = $mEvaluation->getEvaluation($idEvaluation);

if($evaluationList['id_users_evaluator'] != $_SESSION['Zend_Auth']['storage'
    ]['id'] OR $evaluationList['date_real']<>'0000-00-00')
{
   $this->_redirector->gotoSimple('detail',
   'user',
   'people',
   array(
   'user' => $_SESSION['Zend_Auth']['storage']['id'],
   )
   );
}
```

Výpis 4: Kontrola práv přístupu uživatele

1. **addAction**

   Funkce sloužící k vytvoření nového hodnocení zaměstnance. Nejprve se kontroluje, zdali přihlášený uživatel smí hodnotit. Pokud je hodnotitel přímým nadřízeným hodnoceného a zároveň s ním nemá naplánováno žádné jiné hodnocení, je povinen zadat plánované datum hodnocení, které je defaultně nepovinné. Dále je z databáze vytažen kompetenční model, jeho kompetence a hodnotitelné chování dané kompetence. U odeslaných dat se dále kontrolují povinné položky a ostatní vstupy. U sebehodnocení je povinné ohodnotit celou sekci chování kompetencí. Nejedna-li se o sebehodnocení, je povinné vyplnit alespoň jedno z chování kompetence. Navíc lze nastavit povinnost vyplnit celou sekci chování kompetencí, i pokud se nejedná o sebehodnocení. Nevyskytla-li se žádná chyba, funkce pokračuje dále s ukládáním dat. Každé z chování kompetence má svou váhu a pokud ukládáme neúplnou kompetenci je nutné váhu přepočítat. Existují kompetence, u kterých stačí, aby jedno z chování bylo ohodnoceno známkou **1**, aby výsledná známka celé kompetence byla **1**. Dále pak funkce zjišťuje zdali je potřeba naplánovat další hodnocení. Jedná-li se o sebehodnocení, hodnotitel není přímým nadřízeným hodnoceného, případně přímý nadřízený již má s hodnoceným naplánováno další hodnocení, pak není potřeba příští hodnocení plánovat.

2. **detailAction**

   Funkce pro zobrazení detailu již vytvořeného hodnocení. Zkontroluje, zdali přihlášený uživatel má přístup k detailu daného hodnocení, a dále z databáze vytahuje data o hodnocení. Pokud se jedná o kompetenci, které stačí pouze jedna známka s hodnotou **1**, pak bude u všech chování této kompetence hodnota atributu **rank 1**, tedy za předpokladu, že alespoň jedno z chování bylo ohodnoceno známkou **1**. V takovém případě se v detailu hodnocení nepoužije atribut **rank**, ale atribut **rank_original**, který uchovává původní hodnotu ohodnocení daného chování kompetence.

3. **addPlanAction**

   Funkce sloužící k vytvoření plánu hodnocení. Plán hodnocení může vytvářet pouze uživatel, který je v zaměstnanecké struktuře a má pod sebou tým podřízených zaměstnanců. Pokud má uživatel plánující hodnocení přístup do všech větví zaměstnanecké struktury, pak z databáze vytáhneme všechny podřízené tohoto uživatele, v opačném případě vytáhneme pouze přímé podřízené. U seznamu hodnotitelů prvně vybere všechny možné hodnotitele dle **Id** a pokud má některý z nadřízených přístup do všech větví zaměstnanecké struktury, tak seznam doplníme. Dále, pokud má hodnocený zaměstnanec kompetenční model, tak jej pošleme dál do **view**. Nakonec proběhne kontrola vstupů, ověří se že hodnotitel a hodnocený nejsou na stejné úrovni a plán hodnocení se uloží. Pokud navíc hodnocený zaměstnanec má nastavenou hodnotu atributu **self_evaluation_repeat** na **1**, pak se navíc vytvoří i související plán na sebehodnocení.

4. **editPlanAction**
   Funkce pro úpravu plánu hodnocení. Seznam hodnocených a hodnotitelů se naplní podle vstupních parametrů. Stejně jako u funkce **addPlanAction** se ověří vstupy a navíc se funkce pokusí upravit změněné datum i u souvisejícího sebehodnocení. Po uložení je uživatel přesměrován na stránku odkud přišel.

5. **competencyAction**
   Funkce sloužící k zobrazení přehledu hodnocení daného zaměstnance v rámci jeho zařazení do kompetenčního modelu. Zobrazují se všechny kompetence, známky jejich chování a celkové ohodnocení každé kompetence v procentech. Modely, ve kterých nebyl zaměstnanec hodnocen, se nezobrazují.

3.2.4 View

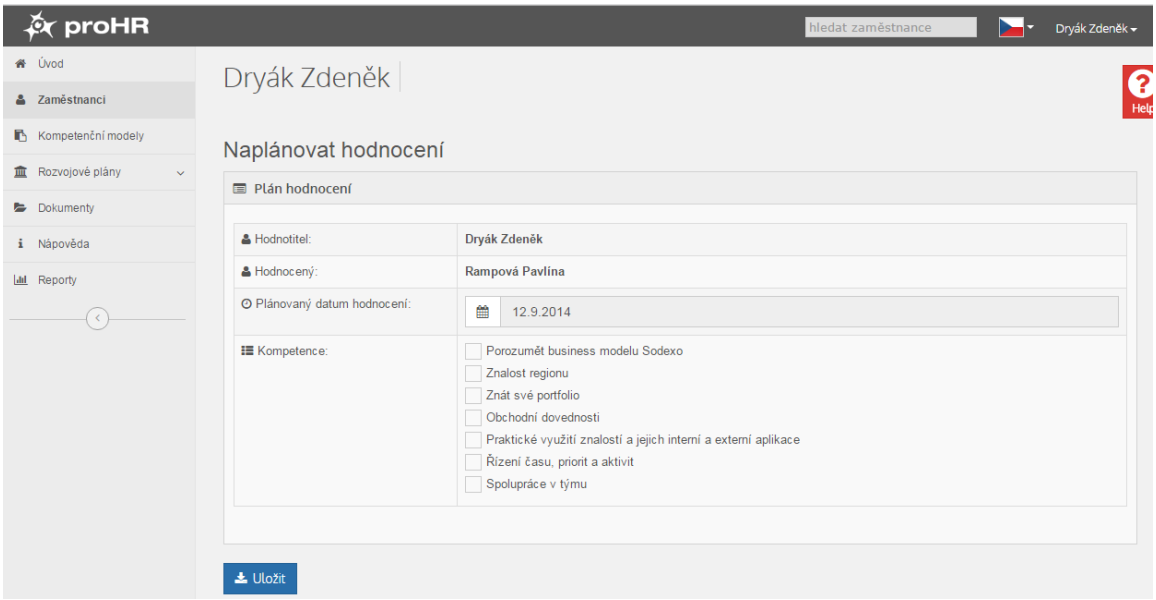
Pro **view** jsem vytvořil celkem pět skriptů které, jak je v Zendu zvykem, odpovídají pěti funkcím naimplementovaným v **controlleru**. Většinu funkcí ve **view** jsem řešil pomocí *PHP* a **JavaScript** jsem použil například při tvorbě kalendáře. Kalendář se vyskytuje ve skriptech **add, addPlan a editPlan** u kterých se zadává například datum hodnocení. Prvky, které jsem například použil, jsou **table-responsive, radio-button, checkbox** a tak podobně.

1. **add**
   V tomto skriptu jsem kromě klasických prvků použil například dynamicky rozbalující se okno **data_toogle**, ve kterém jsou hodnotitelná chování kompetencí.

```
<script type="text/javascript">
$(function(){
var dtp1 = $('#startDate').datepicker({"format": "d.m.yyyy", "autoclose
    ": true, "language":"cs", "weekStart": 1})
.on('changeDate', function(e) {
dtp1.datepicker('hide');
});
});
</script>
```

Výpis 5: JavaScript pro kalendář

2. **competency** V tomto skriptu jsem navíc implementoval zobrazení procentuálního úspěchu a průměrné známky z dané kompetence



Obrázek 5: Ukázka výsledku skriptu **edit_plan**

3.3 Vytvoření grafů pro modul hodnocení

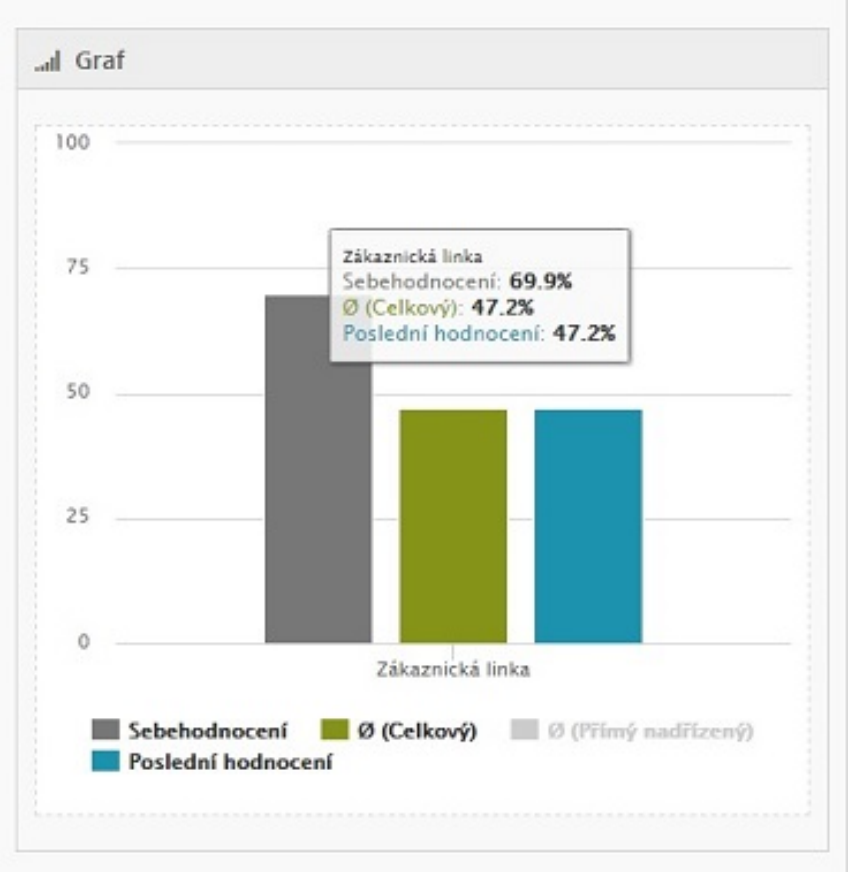
Po dokončení modulu pro hodnocení zaměstnanců jsem dostal nový úkol, a to rozšířit výše zmíněný modul o grafy znázorňující hodnocení jednotlivých zaměstnanců. K vytvoření těchto grafů jsem použil JavaScriptové knihovny Highcharts verze 4.1.7.

3.3.1 Prvotní úpravy

Na začátku řešení tohoto úkolu jsem prvotně musel doplnit model **evaluation_detail** o další funkce spojené s daty pro grafy, například o funkci pro vypočítání průměrné hodnoty hodnocení v procentech pro daného uživatele, kompetenci a přímého nebo naopak nepřímého nadřízeného hodnotitele. Dalším krokem byla úprava controlleru **user**. Controller **user** jsem musel doplnit o zasílání dat do view detailu uživatele pro budoucí grafy.

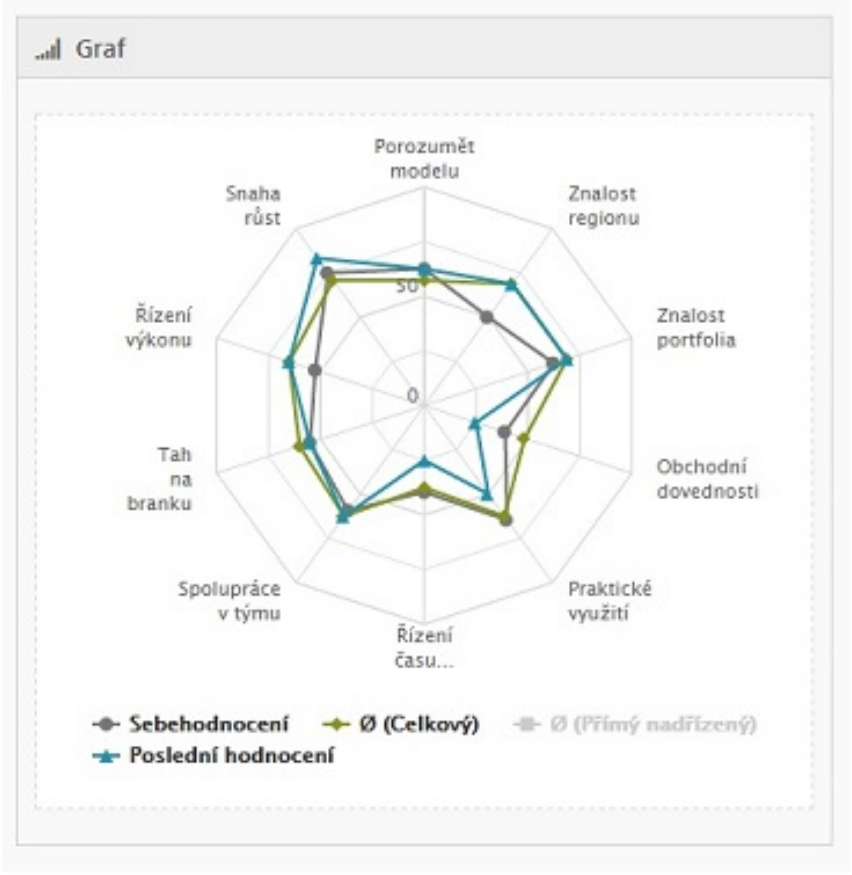
3.3.2 Implementace grafů

Grafy jsem implementoval do view detailu uživatele pomocí výše uvedeného JavaScriptového framerowku *Highcharts*. Výslednou podobu grafů jsem navrhnul ve dvou variantách.



Obrázek 6: Sloupcový graf

- Sloupcový - Pokud má zaměstnanec menší počet kompetencí než tři, pak se hodnocení v grafech vykreslují jako sloupce.
- Přímkový(pavučinový) - U hodnocení s větším počtem kompetencí než tři, se hodnocení v grafech vykresluje pomocí přímek v pavučinovém tvaru. (nastavená hodnota **polar** na **true** a **gridLineInterpolation** na **polygon**)



Obrázek 7: Přímkový graf

Pro oba typy grafů jsem naimplementoval sekci **category** pro souřadnicovou osu x tak, aby na této souřadnicové ose místo čísel byly názvy kompetencí. Pavučinový vzhled druhého typu je docílen nastavením atributu **polar** na hodnotu **true**, což má za výsledek kruhový tvar grafu. Následné nastavení atributu **gridLineInterpolation** na hodnotu **polygon** u souřadnicové osy y, vytváří pavučinový vzhled grafu.

Poslední části JavaScriptu byla sekce **series**, reprezentující přímky a sloupce daných hodnocení. Pro **series** jsem musel naimplementovat plnění daty zasílané controllerem **user**, které jsem musel patřičně rozparsovat pomocí metody **implode** a v poslední řadě nastavit barvu a názvy přímek a sloupců grafů.

```
chart: {
<?php
if(count($this->competency)<3) {
echo "
type: 'column'
";
}
else {
echo "
polar: true,
type: 'line'
";
}
?>
}
```

Výpis 6: Rozlišení mezi sloupcovým a přímkovým grafem

```
   series: [
   {
   name: '<?php echo $this->trans("Sebehodnoceni"); ?>',
   data: [
      <?php
      echo (isset($this->flot_data['self'])) ? implode(', ', $this->flot_data['
          self']) : '';
      ?>
   ],
   pointPlacement: 'on',
   color: '#777777'

   }
```

Výpis 7: Plnění sloupce/přímky daty sebehodnocení

4 Závěr

Práce ve společnosti *proHR Leaders s.r.o.* byla pro mě zcela novou zkušeností. Poprvé jsem měl možnost si vyzkoušet práci v oboru a podílet se na vývoji většího projektu.

Odbornou praxi ve firmě považuji za úspěšnou, vzhledem k novým zkušenostem a znalostem, které jsem v průběhu této praxe získal. Poprvé jsem byl součástí návrhů, konzultací a implementačního řešení v rámci vývoje rozsáhlejší webové aplikace, kterou aplikace *proHR* zajisté je.

Nově jsem si rozšířil obzory o znalosti jazyka *PHP* a jeho frameworku *Zen*, které jsem před začátkem praxe zcela postrádal. Poprvé jsem navrhoval a pracoval s databází v rámci něčeho jiného, než je studium.

Poprvé jsem měl možnost implementačně řešit nový modul pro aplikaci *proHR* pomocí softwarové architektury *MVC* a rozšířit si znalosti o ní.

Dále bych také rád zmínil práci s *JavaScriptovou* knihovnou *Highcharts*, která je skvělým nástrojem pro tvorbu interaktivních grafů, pro obohacení vzhledu webových aplikací a stránek.

Nadále bych rád pokračoval s prací pro firmu proHR Leaders s.r.o., účastnil se dalších projektů a rozšiřoval své znalosti o vývoji webových stránek a aplikací.

Kromě znalostí, které jsem v období odborné praxe získal, bych si také rád do budoucna osvojil práci s dalšími frameworky a jazyky, které jsou v dnešní době stejně jako jazyk *PHP* využívány k tvorbě webových stránek a aplikací.

Reference

[1] *PHP: Hypertext Preprocessor*. URL: http://php.net.

[2] *About - Zend Framework*. URL: http://framework.zend.com/about.

[3] *HTML příručka*. URL: http://www.jakpsatweb.cz/html/.

[4] *CSS Tutorial*. URL: http://www.w3schools.com/css.

[5] *KingAdmin - Responsive Admin Dashboard*. URL: https://wrapbootstrap.com/theme/kingadmin-responsive-admin-dashboard-WB09JXK43.

[6] *JavaScript Tutorial*. URL: http://www.w3schools.com/js.

[7] *Highcharts*. URL: http://www.highcharts.com/.

[8] *About MySQL*. URL: http://www.mysql.com/about.